REMARKS/ARGUMENTS

Claim 30 is cancelled. Claims 1-29 and 31-48 are pending in the application. Reexamination and reconsideration of the application are respectfully requested.

Applicant believes the foregoing amendments comply with requirements of form and thus may be admitted under 37 C.F.R. § 1.116(b)(1) and (b)(2). Alternatively, if these amendments are deemed to touch the merits, admission is requested under 37 C.F.R. § 1.116(b)(3). In this connection, these amendments were not earlier presented because they are in response to the matters pointed out for the first time in the Final Office Action.

Lastly, admission is requested under 37 C.F.R. § 1.116(b)(2) as presenting rejected claims in better form for consideration on appeal.

Claims 22, 23, 27, 28, 30, 38, 40 and 45 were rejected as being anticipated by IPMI, "Intelligent Platform Management Interface Specification" v1.5 (hereinafter IPMI Specification). The rejection as applied to claim 30 is moot in light of the cancellation of that claim. The rejection as applied to the other claims is respectfully traversed.

The rejection make in the final Office Action are substantially identical to those in the non-final Office Action. They have been address in the applicant's response to the non-final Office Action. Therefore, the remarks below focuses on the "Response to Arguments" section of the final Office Action (pages 17-20). The applicant has carefully considered the Examiner's "Response to Arguments" and has carefully studied the IPMI Specification sections cited by the Examiner. However, the applicant still disagree with the Examiner's rejections.

Regarding claim 22, the applicant previously argued:

> The portions of the IPMI Specification cited by the Examiner, 1.6.23 - 1.6.24, discuss platform event filtering and call down lists and alert policies. However, the reference does not teach or suggest "a plurality of message service modules", nor a "message sheet which allows the user to define the corresponding relation between each IPMI message and said message service module." In the IPMI Specification, the configurable alert policies determine how an alert will be processed (see IPMI 1.6.24); they do not determine which one of a plurality of message service modules a message should be sent to for execution, which is required by claim 22.

The Examiner responded (page 17):

> A – IPMI Specification discloses a plurality of service modules [section 1.6.7, paragraph 1]

The applicant respectfully submits that IPMI Specification 1.6.7, paragraph 1 does not disclose a plurality of message service modules. The claimed "message service modules" designates each IPMI message a default execution procedure. IPMI Specification 1.6.7, paragraph 1 states:

> IPMI's extensibility and scalability mean that each platform implementation can have a different population of management controllers and sensors, and different event generation capabilities. The design of IPMI allows system management software to retrieve information from the platform and automatically configure itself to the platform's capabilities. This greatly reduces or eliminates the need for platform-specific configuration of the platform management instrumentation software - enabling the possibility of "Plug and Play" platform-independent instrumentation software.

Clearly, there is no disclosure or suggestion of a message service module in this paragraph, much less the "a plurality of message service modules" and "message sheet which allows the user to define the corresponding relation between each IPMI message and said message service module" required in claim 22.

The applicant would like to emphasize again that the IPMI system of claim 22 requires "a plurality of message service modules". Moreover, to manage the plurality of message service modules, the claimed IPMI system employs "a programmable-configured message sheet which allows the user to define the corresponding relation between each IPMI message and said message service module." This enables the IPMI system to choose a message service module to process a message and to avoid the problems of more than one message service module trying to read or access the same sensor or EEPROM. The cited IPMI specification does not teach or suggest such a feature.

The final Office Action also stated:

> B - IPMI Specification discloses event filter tables that allow a user to define actions based on events from service modules [IPMI Specification, section 1.6.23, paragraphs 1 and 2].

The applicant respectfully disagrees. The IPMP Specification Sec. 1.6.23 discloses Platform Event Filtering:

> The BMC maintains an *event filter* table that is used to select which events trigger an action and which actions to perform. Each time the BMC receives an event message (either externally or internally generated) it compares the event data against the entries in the event filter table. The BMC scans all entries in the table and collects a set of actions to be performed as determined by the entries that were matched.

However, the claimed message sheet must "allow[ ] the user to define the corresponding relation between each IPMI message and said message service module". The Platform Event Filtering of IPMI does not define the relation between IPMI messages and the message service modules. It only defines what actions are to be performed for an event. As explained in paragraph [0006] of the present specification, events and IPMI messages are not the same concept (emphasis added):

> [0006] Most of the BMC micro-controllers of the IPMI integrate with an A/D converter for monitoring voltage, a fan speed counter, and an IO and a bus for a sensor. As a watchdog timer of the IPMI detects a fault of the CPU, BIOS, OS, or application program of the servers, the IPMI provides a platform event filter (PEF) to correct a fault or follow instructions from the operating terminal on fault correction. Moreover, the IPMI automatically provides system status detection of software/hardware of the servers, an event diary log, system rebooting control, automatic alarm for the event, and auto-system control (such as system power off). For example, the BMC micro-controller of the IPMI utilizes an I^2C digital sensor to obtain by polling the measurement of the host system to monitor the system voltage, temperature, and fan speed variations of the remote host system. The IPMI then decides whether the monitored data exceed the default range and sends an I^2C sensing data (an IPMI message) through an intelligent platform management bus (IPMB) or communicates with the host system through a system management bus (SMBus) interface. Any system exception will be immediately recorded in a system event log (SEL) and the IPMI will call for the PEF to find a response action that matches the exception, for instance, cutting off the power supply, re-plugging in the power, rebooting, sending/broadcasting a warning, and so forth.

Thus, Platform Event Filtering is not related to the "plurality of message service modules" and "message sheet" of claim 22. Accordingly, claim 22 is patentable over the IPMI Specification.

Regarding claim 27, the applicant previously argued that the IPMI Specification does not disclose "an operating system (OS) management module having multiple specific mapping functions for communicating with different types of OS, allowing the advanced IPMI system to function with different OS."

The final Office Action responded that

> C – IPMI Specification discloses communication with different OS [IPMI Specification, 1.6.1, third paragraph, "The independent monitoring, logging, and access functions available through IPMI provide a level of manageability built-in to the platform hardware. This can support systems where there is no system management software available for the particular operating system...", section 1.6.2].

The applicant submits, however, that this paragraph does not disclose multiple specific mapping functions for communicating with different types of OS; it merely states that IPMI can work with systems that do not have system management software for the particular OS. It does not explain how the IPMI system works with different OS, i.e., through multiple mapping functions as claimed or through some other method.

The applicant also argued that in addition, IPMI Specification does not disclose "a hardware management module having a plurality of driver units for communicating with different baseboard management controller (BMC), allowing the advanced IPMI system to function in different hardware environments." IPMI Specification 1.6.2 - 1.6.3 does not show such a hardware management module. Although it mentions "connecting addition management controllers to the system using the IPMB", there is no description of a component with a plurality of driver units for the different BMCs.

The final Office Action responded that:

> D - IPMI Specification discloses connecting additional management controllers and the required driver units [IPMI Specification, paragraphs 1.6.2 – 1.6.3 and page 28, table 3-1 and page 31].

The applicant respectfully disagrees. Paragraph 1.6.2 discusses IPMI's relationship to other management standards. Paragraph 1.6.3 discusses management controllers and the IPMB (Intelligent Platform Management Bus). IPMI supports the extension of platform management by connecting additional management controllers to the system using the IPMB. While there may be some driver units for communicating to the controller, such driver units are in the extension platform respectively, not in the IPMI system.

Accordingly, claim 27 is patentable over the IPMI Specification.

Regarding claim 28, the applicant previously argued that the IPMI Specification does not disclose "a memory control unit which regularly poll a new sensing event in the EEPROM of the sensor unit ….". The final Office Action responded:

> E - IPMI Specification discloses polling sensing events [IPMI Specification, page 47, sections 6.10 - 6.10.1].

However, the applicant submits that the IPMI Specification, 6.10 to 6.10.1, page 47, merely describes a Receive Message Queue:

> Messaging between system software and the other management busses, such as the IPMB, is accomplished using channels and a *Receive Message Queue*. A channel is a path through the BMC that allows messages to be sent between the system interface and a given bus or message transport. The Receive Message Queue is used to hold message data for system software until system software can collect it. All channels share the Receive Message Queue for transferring messages to system management software. The Receive Message Queue data contains channel, session, and IPMI addressing information that allows system software to identify the source of the message, and to format a message back to the source if necessary.
>
> System management software is responsible for emptying the Receive Message Queue whenever it has data in it.

Claim 28, on the other hand, requires polling new sensing events in the EEPROM of the sensor unit. The Receive Message Queue in the IPMI Specification does not correspond to the EEPROM of the sensor. The Receive Message Queue receives messages from all channels, while the claimed EEPROM of the sensor unit is a storage for messages from one sensor unit. Therefore, emptying the Receive Message Queue by the system software does not correspond to the claimed polling new sensing events in the EEPROM as claimed in claim 28. Accordingly, claim 28 is patentable over the IPMI Specification.

Regarding claim 38, the applicant previously argued that

> IPMI Specification does not disclose of "by a plurality of programmable-configured message processing units, multi-processing concurrently the IPMI messages, each initiating according to each IPMI message a message service module having a default execution procedure." (Emphasis added.) The Examiner cited paragraphs 1.6.23 and 1.6.25 of the IPMI Specification for teaching this feature. Paragraph 1.6.23 describes Platform Event Filtering; it does not teach or suggests concurrent multi-processing of messages. Paragraph 1.6.25 describes Channel Model; it teaches a plurality of channels for routing IPMI messages, but still does not teach or suggest concurrent multi-processing of messages.

The final Office Action responded:

G - IPMI Specification discloses concurrent multi-processing of messages [IPMI Specification, 1.6.23 (default message procedure) and 1.6.25 (multiple simultaneous sessions)].

However, the multiple simultaneous sessions in Sec. 1.6.25 do not correspond to the multi-processing by a plurality of message processing units. "Sessions" are described in the IPMI Specification as follows:

> Channels can be *session-based* or *session-less*. A *session* is used for two purposes: As a framework for user authentication, and to support multiple IPMI Messaging streams on a single channel. Session-based channels thus have at least one user 'login' and support user and message authentication. Session-less channels do not have users or authentication. LAN and serial/modem channels are examples of session-based, while the System Interface and IPMB are examples of session-less channels.
>
> In order to do IPMI messaging via a session, a session must first be *activated*. The act of activating a session is one of authenticating a particular user. This is accomplished using a 'challenge/response' mechanism, where a challenge is requested using a *Get Session Challenge* command, and the signed challenge returned in an *Activate Session* command.
>
> A session has a *Session ID* that is used for tracking the state of a session. The Session ID mechanism allows multiple sessions to be able to be simultaneously supported on a channel.

Although this describes simultaneously supporting multiple session on a channel, it does not describe multi-processing by a plurality of message processing units. The multiple sessions could be supported by a single processing unit, since the sessions are merely a logical framework for user authentication and for multiple IPMI messaging streams. Accordingly, claim 38 and its dependent claim 40 are patentable over the IPMI Specification.

Regarding claim 45, this method claim is similar to apparatus claim 28 in that it requires "polling regularly by a memory control unit a new sensing event in the EEPROM of the sensor unit." Thus, for the same reasons as explained in connection with claim 28, claim 45 is patentable over the IPMI Specification.

Claims 1-21, 24-26, 31-37, 39, 41-44 and 46-48 were rejected as being obvious over the IPMI Specification in view of Khacherian, U.S. Pat. Appl. Pub. 2003/0063618. This rejection is respectfully traversed.

The applicant previously argued that it would not have been obvious to combine the Khacherian reference with the IPMI Specification. In response, the Examiner argued that "In this case, one having ordinary skill in the art would have knowledge of Khacherian, as such, the combination would have been obvious." The applicant respectfully disagrees. Indeed, knowledge of prior art by one of ordinary skill in the art is presumed in 35 USC §§102 and 103. However, the Office still has to articulate a reasoning for combining prior art in making an

obviousness rejection. See KSR v. Teleflex, 127 S. Ct. 1727, 1741 (2007) (citing In re Kahn, 441 F.3d 977, 988 (Fed. Cir. 2006): "Rejections on obviousness grounds cannot be sustained by mere conclusory statements; instead, there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness.")

In the instant case, the Examiner has not articulated a sufficient reasoning for combining the cited references. Khacherian teaches using memory pointers in a switch fabric. The various components of the switch fabric, namely an Enqueue engine and a Dequeue engine, uses the memory pointer to perform their operations. However, the IPMI is not a switch, and it has different components than the switch fabric of Khacherian. It would not have been obvious as to how the memory pointer would be implemented in the IPMI structure. Thus, the applicant respectfully submits that it would not have been obvious to combine the two cited references.

Accordingly, claims 1 and 31, as well as claims 2-21 and 32-37 that depend thereon, are patentable over the IPMI Specification in view of Khacherian.

Claims 24-26 depend from claim 22, claims 39 and 41-44 depend from claim 38, and claims 46-48 depend from claim 45. As discussed earlier, the IPMI Specification fail to teach or suggest various elements of claims 22, 38 and 45. The Khacherian reference does not cure these deficiencies of the IPMI Specification because Khacherian relates to a switch and does not teach anything about an IPMI system. Accordingly, claims 24-26, 39, 41-44 and 46-48 are patentable over the IPMI Specification in view of Khacherian.

In view of the foregoing, it is respectfully submitted that the application is in condition for allowance. Reexamination and reconsideration of the application, as amended, are requested. If for any reason the Examiner finds the application other than in condition for allowance, the Examiner is invited to call the undersigned attorney at the Los Angeles, California telephone number (213) 625-5076 to discuss the steps necessary for placing the application in condition for allowance.

If there are any fees due in connection with the filing of this response or deficient in fees, please charge the fees to our Deposit Account No. 50-3531.

Respectfully submitted,

Date: Jan. 8, 2009

By: /Ying Chen/
Ying Chen
Registration No. 50,193
Attorney for Applicant(s)

255 S. Grand Ave., #215
Los Angeles, CA 90012
Phone: 213-625-5076
Fax: 213-625-0691